Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (currently amended) A method of automatically generating code to be deployed in an application server, comprising the steps of:

receiving an archive file to be deployed, wherein the archive file includes at least one input class;

introspecting an input class included in the archive file, wherein introspecting the input class included in the archive file includes ~~to~~ automatically generating information relating to the input class by extracting information identifying methods included in the input class; and for each method, extracting information relating to parameters of the method including at least a name and a type of each parameter;

automatically generating a markup language description of the input class based on the extracted information identifying methods included in the input class; and extracted information relating to parameters of the method including at least a name and a type of each parameter;

creating an event handler that leverages multiple templates for a method node found in the markup language description;

registering the event handler;

parsing the markup language description and invoking the registered event handler; and

automatically generating output code using the invoked event handler.

2. (canceled)

3. (canceled)

4. (previously presented) The method of claim 1, wherein the step of automatically generating a markup language description of the input class comprises the step of:

automatically generating an Extensible Markup Language description of the input class based on the extracted information identifying methods included in the input class; and extracted information relating to parameters of the method including at least a name and a type of each parameter.

5. (previously presented) The method of claim 4, wherein the step of creating an event handler comprises the step of:

creating a Simple Application Programming Interface for Extensible Markup Language event handler for a method node found in the Extensible Markup Language description.

6. (previously presented) The method of claim 46, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the Extensible Markup Language description using a Simple Application Programming Interface for Extensible Markup Language parser and invoking the

registered Simple Application Programming Interface for Extensible Markup Language event handler.

7. (original) The method of claim 1, wherein the step of creating an event handler comprises the step of:

creating a plurality of event handlers for a method node found in the markup language description.

8. (original) The method of claim 7, wherein the step of registering the event handler comprises the step of:

registering each of the plurality of event handlers.

9. (original) The method of claim 8, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the markup language description and invoking each of the plurality of registered event handlers.

10. (previously presented) The method of claim 9, wherein the step of automatically generating output code comprises the step of:

automatically generating output code using each of the plurality of invoked event handler in parallel.

11. (canceled)

12. (canceled)

13. (currently amended) The method of claim 10 [[12]], wherein the step of automatically generating a markup language description of the input class comprises the step of:

automatically generating an Extensible Markup Language description of the input class based on the extracted information identifying methods included in the input class; and extracted information relating to parameters of the method including at least a name and a type of each parameter.

14. (previously presented) The method of claim 13, wherein the step of creating a plurality of event handlers comprises the step of:

creating a plurality of Simple Application Programming Interface for Extensible Markup Language event handlers for a method node found in the Extensible Markup Language description.

15. (previously presented) The method of claim 50, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the Extensible Markup Language description using a Simple Application Programming Interface for Extensible Markup Language parser and invoking the

plurality of registered Simple Application Programming Interface for Extensible Markup Language event handlers.

16. (currently amended) A system for automatically generating code to be deployed in an application server comprising:

a processor operable to execute computer program instructions;

a memory operable to store computer program instructions executable by the processor; and

computer program instructions stored in the memory and executable to perform the steps of:

receiving an archive file to be deployed, wherein the archive file includes at least one input class;

introspecting an input class included in the archive file, wherein introspecting the input class included in the archive file includes ~~to~~ automatically generating information relating to the input class by extracting information identifying methods included in the input class; and for each method, extracting information relating to parameters of the method including at least a name and a type of each parameter;

automatically generating a markup language description of the input class based on the extracted information identifying methods included in the input class; and extracted information relating to parameters of the method including at least a name and a type of each parameter ~~s~~;

creating an event handler <u>that leverages multiple templates</u> for a method node found in the markup language description;

registering the event handler;

parsing the markup language description and invoking the registered event handler; and

automatically generating output code using the invoked event handler.

17. (canceled)

18. (canceled)

19. (previously presented) The system of claim 16, wherein the step of automatically generating a markup language description of the input class comprises the step of:

automatically generating an Extensible Markup Language description of the input class based on the extracted information identifying methods included in the input class; and extracted information relating to parameters of the method including at least a name and a type of each parameter.

20. (previously presented) The system of claim 19, wherein the step of creating an event handler comprises the step of:

creating a Simple Application Programming Interface for Extensible Markup Language event handler for a method node found in the Extensible Markup Language description.

21. (previously presented) The system of claim 47, wherein the step of parsing the

markup language description and invoking the registered event handler comprises the step of:

parsing the Extensible Markup Language description using a Simple Application Programming Interface for Extensible Markup Language parser and invoking the registered Simple Application Programming Interface for Extensible Markup Language event handler.

22. (previously presented) The system of claim 21, wherein the step of creating an event handler comprises the step of:

creating a plurality of event handlers for a method node found in the Extensible Markup Language description.

23. (original) The system of claim 22, wherein the step of registering the event handler comprises the step of:

registering each of the plurality of event handlers.

24. (previously presented) The system of claim 23, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the Extensible Markup Language description and invoking each of the plurality of registered event handlers.

25. (previously presented) The system of claim 24, wherein the step of automatically generating output code comprises the step of:

automatically generating output code using each of the plurality of invoked event handler in parallel.

26. (canceled)

27. (canceled)

28. (currently amended) The system of claim 25 [[27]], wherein the step of automatically generating a markup language description of the input class comprises the step of:

automatically generating an Extensible Markup Language description of the input class based on the extracted information identifying methods included in the input class; and extracted information relating to parameters of the method including at least a name and a type of each parameter.

29. (previously presented) The system of claim 28, wherein the step of creating a plurality of event handlers comprises the step of:

creating a plurality of Simple Application Programming Interface for Extensible Markup Language event handlers for a method node found in the Extensible Markup Language description.

30. (previously presented) The system of claim 51, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the Extensible Markup Language description using a Simple Application Programming Interface for Extensible Markup Language parser and invoking the plurality of registered Simple Application Programming Interface for Extensible Markup Language event handlers.

31. (currently amended) A computer program product for generating code to be deployed in an application server comprising:

a computer readable storage medium; and

computer program instructions, recorded on the computer readable storage medium, executable by a processor, for performing the steps of:

receiving an archive file to be deployed, wherein the archive file includes at least one input class;

introspecting an input class included in the archive file, wherein introspecting the input class included in the archive file includes ~~to~~ automatically generating information relating to the input class by extracting information identifying methods included in the input class; and for each method, extracting information relating to parameters of the method including at least a name and a type of each parameter;

automatically generating a markup language description of the input class based on the extracted information identifying methods included in the input class;

and extracted information relating to parameters of the method including at least a name and a type of each parameter;

creating an event handler that leverages multiple templates for a method node found in the markup language description;

registering the event handler;

parsing the markup language description and invoking the registered event handler; and

automatically generating output code using the invoked event handler.

32. (canceled)

33. (canceled)

34. (previously presented) The computer program product of claim 31, wherein the step of automatically generating a markup language description of the input class comprises the step of:

automatically generating an Extensible Markup Language description of the input class based on the extracted information identifying methods included in the input class; and extracted information relating to parameters of the method including at least a name and a type of each parameter.

35. (previously presented) The computer program product of claim 34, wherein the step of creating an event handler comprises the step of:

creating a Simple Application Programming Interface for Extensible Markup Language event handler for a method node found in the Extensible Markup Language description.

36. (previously presented) The computer program product of claim 48, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the Extensible Markup Language description using a Simple Application Programming Interface for Extensible Markup Language parser and invoking the registered Simple Application Programming Interface for Extensible Markup Language event handler.

37. (previously presented) The computer program product of claim 31, wherein the step of creating an event handler comprises the step of:

creating a plurality of event handlers for a method node found in the markup language description.

38. (original) The computer program product of claim 37, wherein the step of registering the event handler comprises the step of:

registering each of the plurality of event handlers.

39. (original) The computer program product of claim 38, wherein the step of parsing the markup language description and invoking the registered event handler comprises the

step of:

parsing the markup language description and invoking each of the plurality of registered event handlers.

40. (previously presented) The computer program product of claim 39, wherein the step of automatically generating output code comprises the step of:

automatically generating output code using each of the plurality of invoked event handler in parallel.

41. (canceled)

42. (canceled)

43. (currently amended) The computer program product of claim 40 [[42]], wherein the step of automatically generating a markup language description of the input class comprises the step of:

automatically generating an Extensible Markup Language description of the input class based on the extracted information identifying methods included in the input class; and extracted information relating to parameters of the method including at least a name and a type of each parameter.

44. (previously presented) The computer program product of claim 43, wherein the step of creating a plurality of event handlers comprises the step of:

creating a plurality of Simple Application Programming Interface for Extensible Markup Language event handlers for a method node found in the Extensible Markup Language description.

45. (previously presented) The computer program product of claim 49, wherein the step of parsing the markup language description and invoking the registered event handler comprises the step of:

parsing the Extensible Markup Language description using a Simple Application Programming Interface for Extensible Markup Language parser and invoking the plurality of registered Simple Application Programming Interface for Extensible Markup Language event handlers.

46. (previously presented) The method of claim 5, wherein the step of registering the event handler comprises the step of:

registering the created Simple Application Programming Interface for Extensible Markup Language event handler for a method node found in the Extensible Markup Language description.

47. (previously presented) The system of claim 20, wherein the step of registering the event handler comprises the step of:

registering the created Simple Application Programming Interface for Extensible Markup Language event handler for a method node found in the Extensible Markup Language description.

48. (previously presented) The computer program product of claim 35, wherein the step of registering the event handler comprises the step of:

registering the created Simple Application Programming Interface for Extensible Markup Language event handler for a method node found in the Extensible Markup Language description.

49. (previously presented) The computer program product of claim 44, wherein the step of registering each of the plurality of event handlers comprises the step of:

registering the plurality of created Simple Application Programming Interface for Extensible Markup Language event handlers for a method node found in the Extensible Markup Language description.

50. (previously presented) The method of claim 14, wherein the step of registering each of the plurality of event handlers comprises the step of:

registering the plurality of created Simple Application Programming Interface for Extensible Markup Language event handlers for a method node found in the Extensible Markup Language description.

51. (previously presented) The system of claim 29, wherein the step of registering each of the plurality of event handlers comprises the step of:

registering the plurality of created Simple Application Programming Interface for Extensible Markup Language event handlers for a method node found in the Extensible Markup Language description.